



Lesson 1

Neural Network Parameters: Weights and Biases

Neural Network Parameters

- |
 - |— Weights
 - | |— Adjust connection strength
 - | |— Initialized randomly or strategically
 - | |— Updated via gradient descent
 - |— Biases
 - | |— Shift activation function
 - | |— Initialized randomly
 - | |— Adjusted to minimize error
- |— Non-Linear Functions
 - | |— Introduce non-linearity
 - | |— Model complex relationships
 - | |— Examples: ReLU, Sigmoid, Tanh

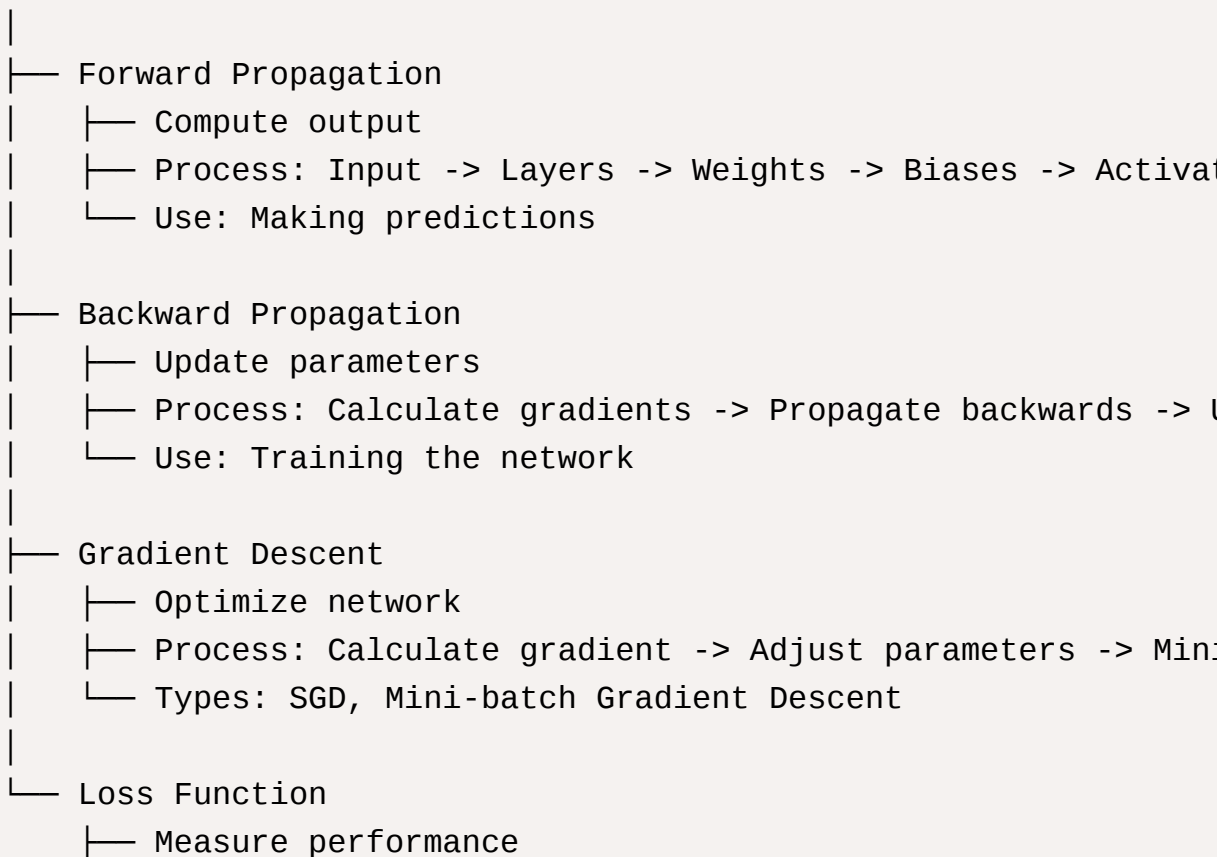
Quick Reviewer

- **Weights**
 - **Adjust Strength:** Modify the influence of one neuron on another.
 - **Initialization:** Random or planned.
 - **Learning:** Optimized during training using gradient descent.
- **Biases**

- **Shift Activation:** Allows for better fitting of the model by adjusting the function.
 - **Initialization:** Random.
 - **Learning:** Refined during training to improve accuracy.
 - **Non-Linear Functions**
 - **Purpose:** Add complexity to the model.
 - **Impact:** Enable the network to learn and represent intricate patterns.
 - **Examples:** ReLU (Rectified Linear Unit), Sigmoid, Tanh.
-

Neural Network Training: Forward Propagation vs. Backward Propagation

Neural Network Training



- └ Quantifies error between predictions and actual values
- └ Guides optimization to reduce error

Quick Reviewer

- **Forward Propagation**

- **What:** Computes network output.
- **How:** Data → Layers → Weights/Biases → Activation Functions.
- **When:** For making predictions.

- **Backward Propagation**

- **What:** Updates weights and biases.
- **How:** Calculate gradients → Backward propagation → Update parameters.
- **When:** During training to minimize error.

- **Gradient Descent**

- **What:** Optimization algorithm.
- **How:** Compute gradient → Adjust parameters → Minimize loss.
- **Goal:** Improve model accuracy.

- **Loss Function**

- **What:** Measures prediction accuracy.
 - **How:** Quantifies difference between predicted and actual values.
 - **Role:** Guides training to reduce errors.
-